The Author-Topic Model and the author prediction

Jiasi Song
Computer Science Department of UW-Madison
sindgers@cs.wisc.edu

Abstract

The author-topic model is a generative model for documents that extends Latent Dirichilet Allocation to include authorship information, which is proposed by Michal Rosen-Zvi et al. The model connects each author to a multinomial distribution over topics and associated each topic with a words' multinomial distribution. A document with multiple authors is modeled as a distribution over topics that are a mixture of the distributions associated with the authors.

In this project, I re-implement the model to a collection of about 250 NIPS conference papers (be chosen randomly from a collection of about 1700 NIPS papers). Exact inference is intractable for these datasets and I use Gibbs sampling to estimate the topic and author distributions. The tagging results with different topic numbers are given.

After getting the distribution values, I present a new method that apply maximum likelihood estimate to do author prediction on about other 100 papers of which the authors are in the same set as the training papers. The precision of prediction is given.

Key word: author-topic model; Gibbs sampling; multinomial distribution; tagging; author prediction;

Introduction

As there are more and more Web and various specialized digital libraries, it is of increasingly greater importance to automatically do the extraction of information we need from text. And when we are searching some information of a specific area, we met a anonymous article and we want to find the author of it. This project is presenting a method to do author prediction.

Before we do prediction, we need to tag the papers. Tagging of documents is a classical problem addressed in statistical natural language processing and machine learning. The tagging result, a characterization of document content can be used to organize, classify, or search a bunch of documents. It is a common method to use generative models for documents and extract topic-based content representations, modeling each document as a mixture of probabilistic topics (e.g., Blei, Ng, & Jordan, 2003; Hofmann, 1999). However, most research in this area just pays attention to the problem of attributing the authorship by superficial features.

The algorithm proposed by Michal Rosen-Zvi et al in the project explores both the hidden topics in large collection of text document and models the way each author of documents applies those topics in their papers. And by the connections among authors, topics and words extracted by the author-topic model, we can predict the author list of a new paper as long as we have large enough data set.

The paper is organized as follows. In next section, we introduce author-topic model and Gibbs sampling method, section 3 is devoted to describing how to predict authors based on the section 2's results. Section 4 shows the experimental results of the algorithm and section 5 is the conclusion.

Author-topic Model and Gibbs Sampling

In the author-topic model, each document is represented by bag-of-word; it models both the document contents and the author's interests by a topic-based representation. Like the Fig.1 shows, the document d is written by a group of authors, a_d, but each word w of the document is only written by one author, which we choose it randomly. Then, according to the author's interest distribution θ, we pick a topic. And the word is generated from the picked topic based on the multinomial distribution Φ over words of that topic.

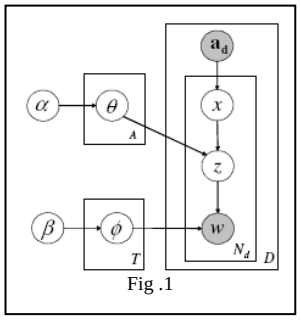

Fig .1

The Gibbs sampling is a method we used to estimate θand Φ in the author-topic method. The algorithm is begun

with uniform initialization of θ_0 and Φ_0. Then it executed multiple iterations. In each iteration, it randomly generates words of each document according to current θ_i and Φ_i , and using the new generated documents to update θ_{i+1} and Φ_{i+1} .

Author Prediction

After getting the values of θ and Φ by Gibbs Sampling, I was trying to use maximum likelihood estimate to do author prediction.

First, I calculate the most probable author of each word by:

$$\arg\max_{a_i} \log(\sum_{j=1}^{T} \theta_{a_i j} \phi_{j w_i}) \quad (1)$$

Where a_i is the author of word w_i, and there are T topics, with $\theta_{ai,j}$ indicating the possibility that how much percents of author a_i's interest is in topic j and $\Phi j,w_i$ is the percentage of word w_i appears in the topic j.

Then I find the top authors which have been chosen as the author of words in the document for most times, and make them the prediction of author.

The second method I tried of prediction is assuming every word of the document are all written by one author, then calculate the possibility of each author to write the whole document, list the top ones as the predicted authors.

The equation of finding the authors with largest probability of writing the whole document D is below:

$$\arg\max_{a_i} \log(\sum_{W_i \in D} \sum_{j=1}^{T} \theta_{a_i j} \frac{\phi_{j w_i}}{\sum_{j' \in T} \phi_{j' w_i}}) \quad (2)$$

There is another difference between two approaches: in the second method, I used the percentage indicated the possibilities of the word i representing topic j instead of the topic j will use word i, of which the former possibility is also can be thought as the possibility of the word is rendering topic j by the specific author.

Experiment Results

The dataset of the experiment is from a pre-processed dataset of NIPS papers[4], which contains the NIPS conference papers from 1987 to 1999. Because the Gibbs sampling method is time consuming, thus I make a random subset of the above dataset and do the experiment. The subset contains 243 papers, 387 authors and use V=11848 unique words.

Then I did two groups of experiments, both experiments are executed 2000 iterations.

In the first experiment, #topic is 20, and the table below are the most correlated words of each topic.

Table 1: the most correlated words for 20 topics

1: output data unit processing feature training speech random science ieee	2: units class line step layer density present examples phase computation	3: system networks field time learning functions systems experiments general computer	4: signal vectors noise model target input features patterns simulation variable	5: data neural inputs small weights properties dimensional classifier cell connections
6: network hidden paper probability weight frequency figure recurrent distributed made	7: algorithm networks approach systems activity motion statistical components case distance	8: set information image parameters rate threshold states architecture log dynamic	9: state fig order optimal cells map form maximum parallel sample	10: model results algorithm problem gaussian component terms computational required training
11: system direction network number velocity energy fixed back human regions	12: neural performance node work models decision function signals obtained independent	13: values number control method information matrix defined constant gradient applied	14: time learning recognition set term note theory classification regression theorem	15: network local net single analog average simple chip center analysis
16: figure training networks space similar sequence solution stochastic table found	17: based distribution trained process representation orientation time images synaptic network	18: linear shown neurons function point pattern rule references due response	19: function nodes neuron visual test section vector current problems activation	20: input error large algorithms learning layer left cells weights standard

There are many overlaps among the topics' representative words, and most of them cannot make the topic clear. Thus I modified #topics to 10, and implemented it again.

Table 2: The most correlated words for 10 topics

1: results neural system control distribution neurons probability matrix shows method	2: number learning model fig field form system node dimensional task	3: network networks time layer performance training data figure vectors representation	4: learning output state error rate size direction classification random work	5: network weight case recognition functions neural nodes small neuron single
6: algorithm network systems problem weights processing cells approach pattern based	7: data neural units models time point paper trained algorithms set	8: input information set vector hidden shown high research speech defined	9: function figure linear space networks parameters visual signal threshold net	10: model training unit image large simple local current process values

Table 3: The most correlated authors for 10 topics

1: Weber_Weber Raysz_J Toomarian_N Downs_O Kirk_D	2: Naillon_M Fokoue_E Uno_Y Gyorgyi_G Madarasmi_S	3: Sudbrak_T Ma_S Hari_R Huang_W Kosaka_H	4: Mumta_N Buckland_K Troyansky_L Kremer_S Winter_C	5: Saad_D Hush_D Townsend_J Marchand_M de-Gerlache_M
6: Indiveri_G Annaswamy_A Pazzani_M Shashua_A Cardin_R	7: Luttges_M Erel_J Fallside_F Vigario_R Saad_D	8: Courchesne_E Hirschman_L Levy_N Robinson_A van-den-Bosch_A	9: Obradovic_Z Micchelli_C Roychowdhury_V Kompe_R Kotani_K	10: Bennett_K Allman_J Gluck_M Fox_G Burnod_Y

The word tagging result looks better, but still cannot make a clear sense for some topics

Then based on the above experiment results (10 topics), the two methods of author prediction were executed. I used other 100 papers of which the authors are in the same set as the training papers and did author prediction on them. In these 100 papers, 65 of them are written by only one author. And because of the time limit, I only did one group of experiments.

Here we use a similarity measure to evaluate the prediction authors' precision:

$$\frac{1}{2N-1}\left(\begin{array}{l} N(100-abs(1-\arg_{Prediction}(a_1))) \\ +\sum_{i=2}^{N}(100-abs(i-\arg_{Prediction} a_i)) \end{array}\right) \quad (3)$$

Where N is #authors of the document, $\arg_{Prediction}(a_i)$ is the order of i_{th} author of the document in the prediction result. If the prediction is right, then $\arg_{Prediction}(a_i) = i$. Because the first author did most work in the paper, I give the first author's prediction precision a much bigger weight than others. By this similarity measure, perfect prediction will get 100, while the totally wrong prediction gets 1.

Below are the results of two prediction methods:

Method	Total Precision	Precision of 1 author paper	Precision of >1author paper
1	9.0157	6.0769	14.7309

2	78.9645	95.1077	51.2403

For the method 2, 58 papers out of 100 papers get the perfectly correct predictions, all of which are with only one author.

Conclusion

In the paper, I re-implement the author topic model and I proposed 2 new methods to do author prediction.

The author-topic model can explore relationship between authors, documents and words. To have the topic tagging make sense, it's important to choose the proper topic number.

And my first method of author prediction seems unsuccessful, while the second one is much better, especially in prediction of papers with only one author.

However, the time complexity of the algorithm is too high, which takes about 18 hours to do 2000 iterations of Gibbs sampling. And because of the time limit, I didn't implement other methods of prediction to compare with my methods, which may make the good performance of method 2 not that convictive. Thus, future work can be done about a more convictive evaluation of the prediction method, and acclerate the algorithm.

References

1. *Finding scientific topics*, T. Griffiths and M. Steyvers, Proceedings of the National Academy of Sciences, 2004

2. *The author-topic model for authors and documents*, M. Rosen-Zvi, T. Griffiths, M. Steyvers, P. Smyth, Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence, 2004.

3. *Probabilistic author-topic models for information discovery* M. Steyvers, P. Smyth, M. Rosen-Zvi, T. Griffiths, Proceedings of the Tenth ACM SIGKDD Conference, 2004.

4. http://www.cs.toronto.edu/~roweis/data.html

5. A. McCallum (1999). *Multi-Label Text Classification with a Mixture Model Trained by EM*. AAAI'99 Workshop on Text Learning.

6. Blei, D. M., Ng, A. Y., and Jordan, M. I., (2003) *Latent Dirichlet allocation*, Journal of Machine Learning Research 3, pp. 993-1022.

7. W. Gilks, S. Richardson, D. Spiegelhalter (1996). *Markov Chain Monte Carlo in Practice*. Chapman and Hall.

8. Hofmann, T. (1999) *Probabilistic latent semantic indexing*, in Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR'99).